Class 5: Agenda, Questions, and Links

1 Warm-Up

Suppose you roll a 6-sided coin n times. Use a Chernoff bound to bound the probability that you see more than $\frac{1+\delta}{6} \cdot n$ threes, where $\delta \in (0,1)$. What bound do you get as a function of n? [When prompted, share your response in the text responses box.]

2 Announcements

- HW3 is posted, due in a week.
- Monday's class will be in-person, in a whiteboard-equipped the tree-well on the engineering quad!

3 Questions?

Any questions from the minilectures or warmup? (Moment generating functions; Chernoff bounds)

4 Randomized Routing

[Slides with setup; the summary is below]

The problem we tackle will lead to a concrete solution to the following important challenge: Suppose we want to design a network with M nodes and a routing protocol in such a way that 1) we have relatively few edges in the network (ie $O(M)$ or $O(M \log M)$), and 2) if each node has a message to send to a some other node, the messages can all be routed to their destinations in a timely manner without too much congestion on the edges.

- Let H be the n-dimensional hypercube. There are 2^n vertices, each labeled with an element of $\{0,1\}^n$. Two vertices are connected by an edge if their labels differ in only one place. For example, 0101 is adjacent to 1101.
- Each vertex i has a packet (also named i), that it wants to route to another vertex $\pi(i)$, where $\pi : \{0,1\}^n \to \{0,1\}^n$ is a permutation.

- Each edge can only have one packet on it at a time (in each direction). Time is discrete (goes step-by-step), and the packets queue up in a first-in-first-out queue for each (directed) edge.

4.1 Group work: Bit-fixing scheme

Consider the following *bit-fixing scheme:* To send a packet i to a node j, we turn the bitstring i into the bitstring j by fixing each bit one-by-one, starting with the left-most and moving right. For example, to send

$$i = 001010$$

to

$$j = 101001,$$

we'd send

$$i = 001010 \to 101010 \to 101000 \to 101001 = j.$$

Group Work

1. Suppose that every packet is trying to get to $\vec{0}$ (the all-zero string). (Yes, I know that this isn't a permutation). Show that if every packet used the bit-fixing scheme (or, any scheme at all) to get to its destination, the total time required is at least $(2^n - 1)/n$ steps.

   ***Hint**: How many packets can arrive at $\vec{0}$ at any one timestep? How many packets need to arrive there?*

2. Suppose that n is even. Come up with an example of a permutation π where the bit-fixing scheme requires at least $(2^{n/2} - 1)/(n/2)$ steps.

   ***Hint**: Consider what happens if $(\vec{a}, \vec{b})$ wants to go to $(\vec{b}, \vec{a})$, where $\vec{a}, \vec{b} \in \{0,1\}^{n/2}$, and use part 1.*

   At this point please go register your progress in the numbers poll.

3. If you still have time, think about the following: what happens if each packet i wants to go to a *uniformly random* destination $\delta(i)$, under the bit-fixing scheme? Will it be as bad as the scheme you came up with in part 2? Or will things it finish in closer to $O(n)$ steps?

4.2 A useful lemma

[Slides. The slides state the following lemma.]

Lemma 1. *Let $D(i)$ denote the delay in the i'th packet. That is, this is the number of timesteps it spends waiting.*

Let $P(i)$ denote the path that packet i takes under the bit-fixing map. (So, $P(i)$ is a collection of directed edges).

Let $N(i)$ denote the number of other packets j so that $P(j) \cap P(i) \neq \emptyset$. That is, at some *point j wants to traverse an edge that i also wants to traverse, in the same direction, although possibly at some other point in time.*

Then $D(i) \leq N(i)$.

Group Work

Let $\delta : \{0,1\}^n \to \{0,1\}^n$ be a completely random function (not necessarily a permutation). That is, for each i, $\delta(i)$ is a uniformly random element of $\{0,1\}^n$, and each $\delta(i)$ is independent.

In this group work, you will analyze how the bit-fixing scheme performs when packet i wants to go to node $\delta(i)$.

Fix some special node/packet i. Let $D(i)$ and $P(i)$ be as above. Fix $\delta(i)$ (and hence $P(i)$, since we have committed to the bit-fixing scheme). But let's keep $\delta(j)$ random for all $j \neq i$. (Formally, we will condition on an outcome for $\delta(i)$; since $\delta(i)$ is independent from all of the other $\delta(j)$, this won't affect any of our calculations).

Let X_j be the indicator random variable that is 1 if $P(i)$ intersects $P(j)$.

1. Assume that we are using the bit-fixing scheme. Argue that $\mathbb{E}[\sum_j X_j] \leq n/2$.

   Hint: *In expectation, how many directed edges are in all of the paths $P(j)$ taken together (with repetition)? Show that this is at most $2^n \cdot n/2$. Then argue that the expected number of paths $P(j)$ that any single directed edge e is in is $1/2$. Finally, bound $\sum_j X_j \leq \sum_{e \in P(i)}$ (number of paths $P(j)$ that e is in) and use linearity of expectation and the fact that $|P(i)| \leq n$ to bound $\mathbb{E}[\sum_j X_j]$.*

   At this point please fill out your progress in the numbers poll.

2. Use a Chernoff bound to bound the probability that $\sum_j X_j$ is larger than $10n$.

   At this point please fill out your progress in the numbers poll.

3. Use your answer to the previous question to bound the probability that the bit-fixing scheme takes more than $11n$ timesteps to send every packet i to $\delta(i)$, assuming that the destinations $\delta(i)$ are completely random.

   Hint: *Lemma 1.*

   At this point please fill out your progress in the numbers poll.

   If you still have time, think about the following:

4. However, the destinations are not random! They are given by some worst-case permutation π. Using what you've discovered above for random destinations, develop a randomized routing algorithm that gets every packet where it wants to go, with high probability, in at most $22n$ steps.

   Hint: *The fact that $22n$ is twice of $11n$ is not an accident.*

4.3 Going over solutions, and coming up with the final algorithm!

Before we go over solutions, let's see if we can figure out the final algorithm, assuming that it if the destinations are random, the bit-fixing algorithm works w.h.p. in time $O(n)$. How should we perform the routing to achieve $O(n)$ expected time, even for a worst-case permutation?

[In remaining time, go over solutions and answer questions]